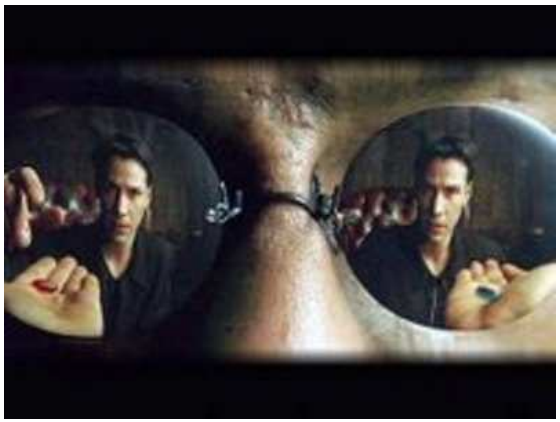
Integrity Virtual Machines

CGO – EPIC7, April 2008

Christophe de Dinechin
Architect, Integrity Virtual Machines

Virtualization?



Virtualization Key Points

Job description: “I’m building the Matrix”

- Virtualization is about choice
- It disconnects you from reality
- It grants you new capabilities
- It sometimes introduces glitches

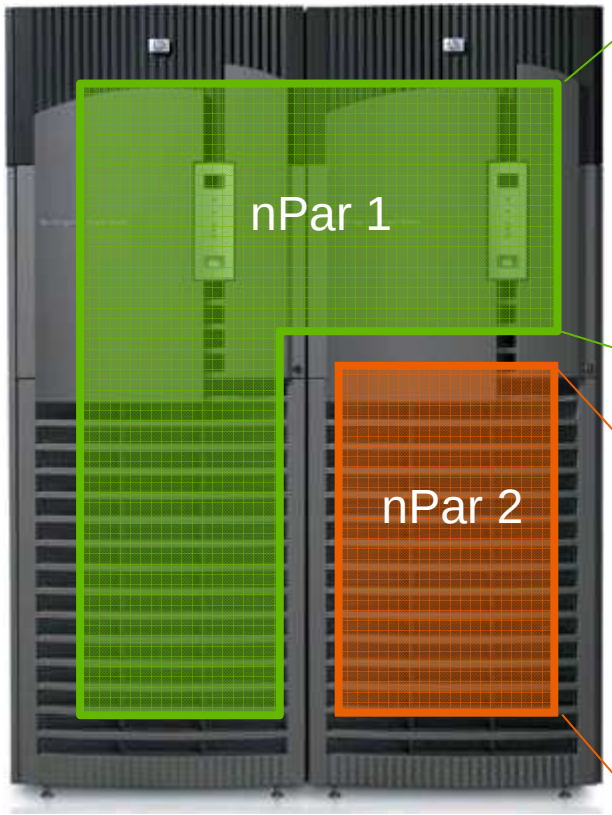
HP Virtualization Overview

You mean we don't do it just for fun?

- For HP customers, it's mostly about manageability
  - Focus is on management software and infrastructure
- It's part of a large ecosystem of solutions
  - Partitioning, workload and system management
  - Hardware: Blades, virtual connect, storage
- It's a key business enabler and differentiator
  - Consolidation, flexibility, cost optimization, accounting

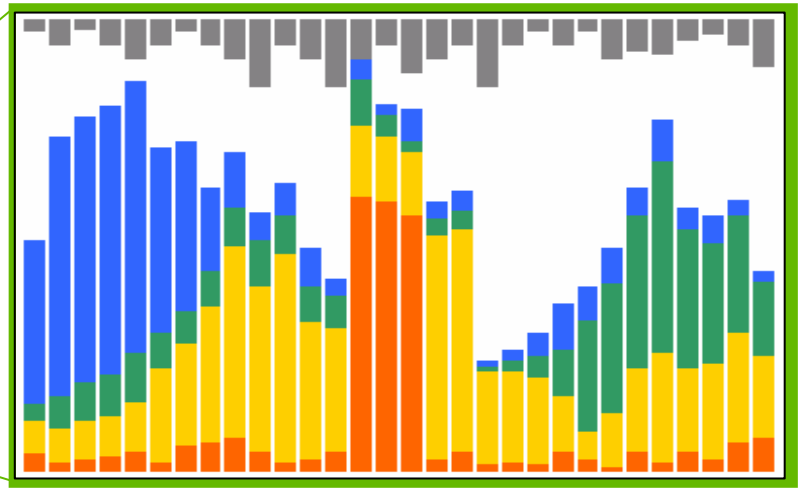
Typical HP Partitioning Ecosystem

From Electrical Isolation to Software Isolation



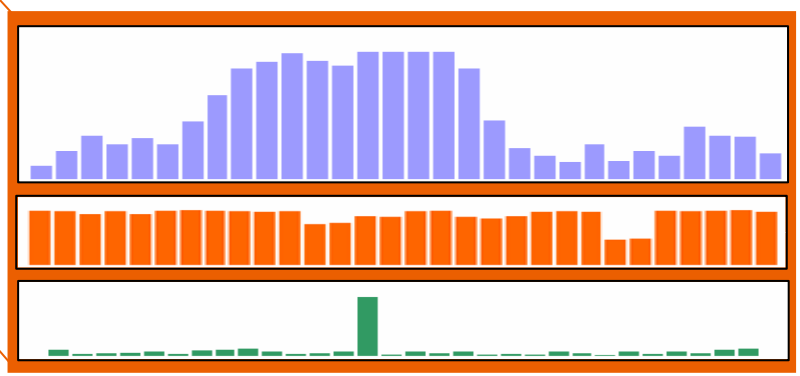
Hardware partitioning (nPar)

Electrically isolated domains



Virtual machines (HPVM)

Some overhead, whole system utilized for workloads



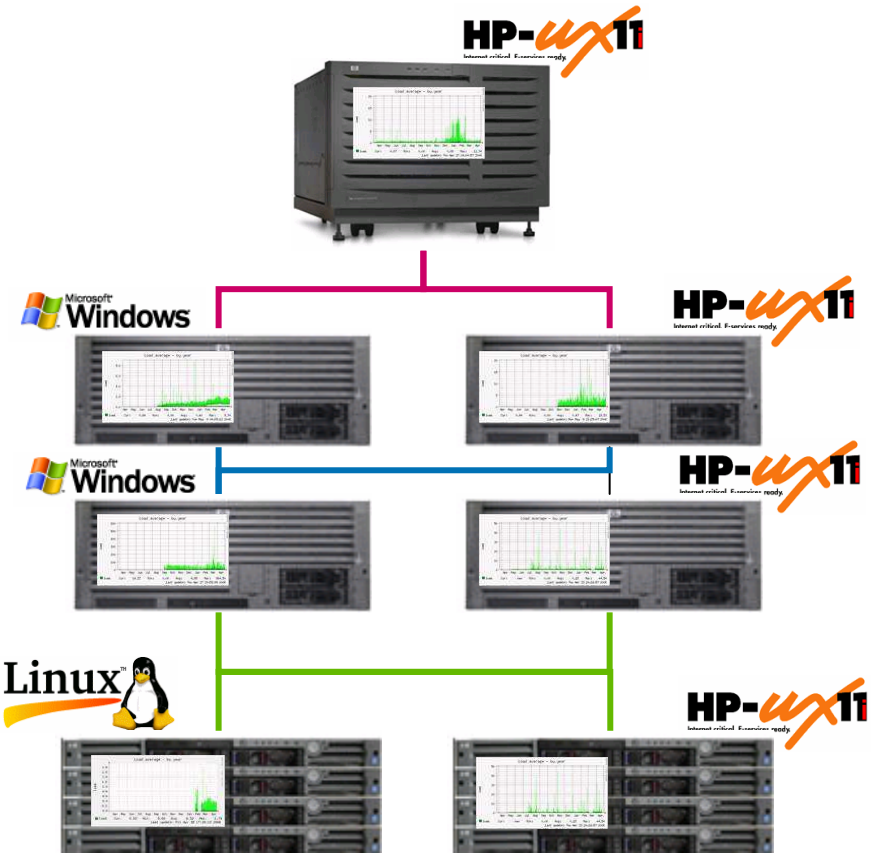
Virtual partitioning (vPar)

No overhead, workload use dedicated resources

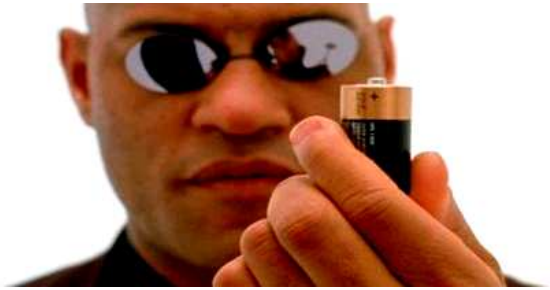
Typical Virtualization Scenario

It's like Play-Doh with Servers

Turn this:

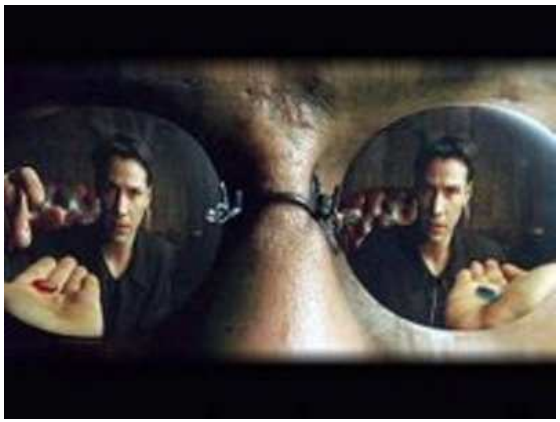


Into this:



Prod: Consolidate underutilized servers
R&D: Simulate complex environments
Support: Replicate all customer cases

A Brief History of HPVM



The early days

Sounds great, but will it work?

- 2000: One engineer project, “MS Word on HP-UX”
- 2001: Advocate 64-bit VM, project cancel/restart
- 2002-2003: Skunkworks Prototype (3 engineers)
  - 3/2002: Self-virtualizing HP-UX boots on simulator
  - 4/2002: … then on real hardware
  - 5/2002: >50% overhead running SDET… ouch
  - 8/2002: Boots EFI shell (firmware user interface)
  - 9/2002: Boots single-user HP-UX
  - 10/2002: Boots multi-user HP-UX
  - 12/2002: Networking
  - 3/2003: Linux 2.4 kernel boots
  - 4/2003: Terrible SpecWeb 1-byte benchmark results ☹
  - 7/2003: First boot of SMP guest

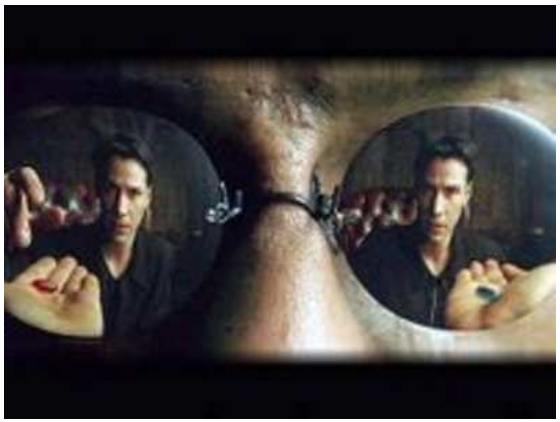
From prototype to product

Hey, it boots, what else do you need?

- End 2003: We get a real team
- 2005: Release 1.0 and 1.2 support HP-UX guests
- 2006: Release 2.0 supports Windows guests
  - As well as burners, tapes, VLAN
- 2007: Release 3.0 supports Linux (RedHat EL 4)
- 2007: Release 3.5 supports SuSE, accelerated I/O

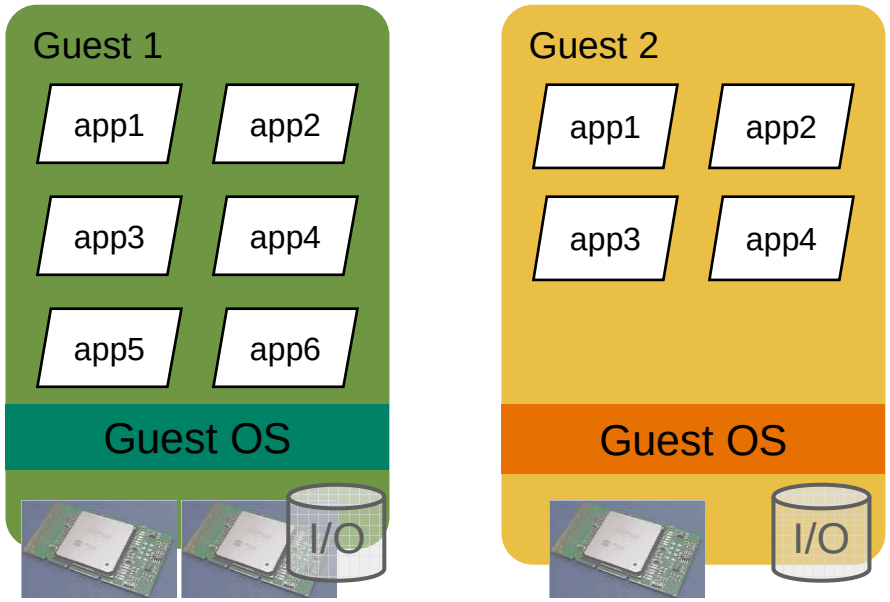
- Currently in the works:
  - Support of HP-UX 11.31 (v3) host, IPv6, new platforms
  - On-line guest migration and OpenVMS “demo” capability
  - Better scalability (8-way guests or more)

Architectural Overview



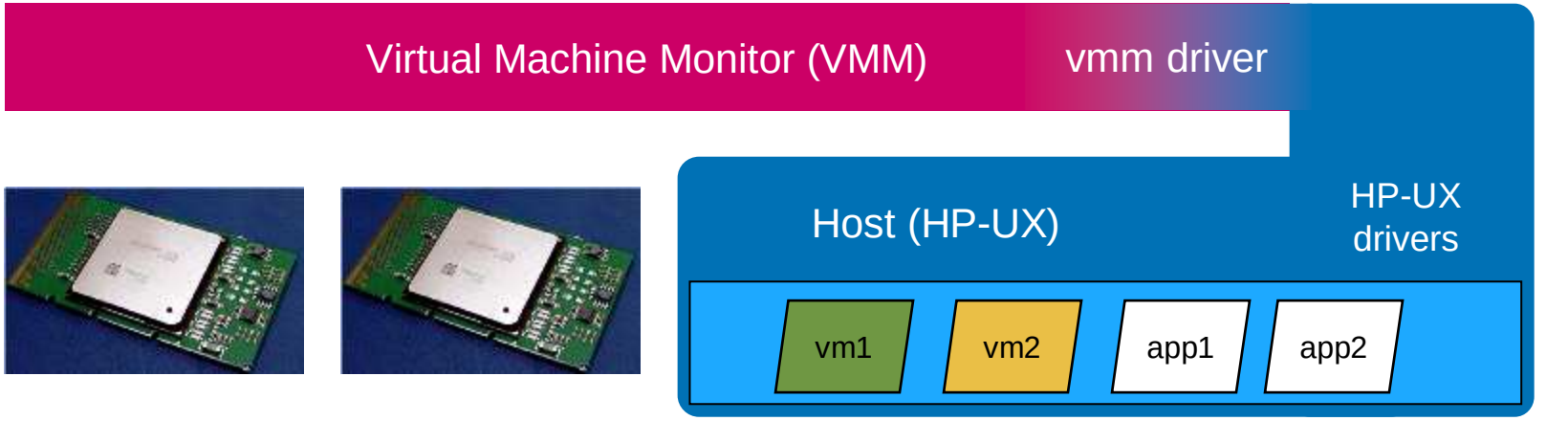
HPVM Technology Overview

Block Diagrams Make it Look So Simple



- A relatively classical hypervisor
- "Hosted", but started as hostless
- Itanium is not fully virtualizable

- Optimized for servers
- Relatively low maintenance cost



Revoking Kernel Privileges

Building “A Prison For Your Mind”

- Operating Systems Expect Privileges
  - Run “privileged operations”, expect “privileged mode”
- The Kernel Would Kill if It Had Privileges
  - Kernels are not ready to share memory, I/O devices,…
- The Kernel Would Die if It Knew…
  - Kernels don’t deal with privilege faults gracefully
- Conclusion: Reduce Privilege Transparently
  - Trap risky operations and emulate them
  - And then: We can reallocate resources!

Required Functional Blocks

There is No Such Thing as a Little White Lie

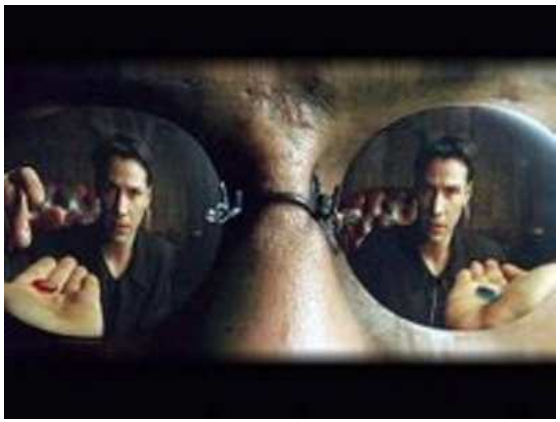
- Memory Management
  - Convert between Host Physical and Guest Physical
- Instruction Emulation
  - Intercept privileged and privilege-sensitive instructions
- I/O Device Emulation
  - Virtual LAN, disks, console, ...
- Firmware & Platform Emulation
  - Virtual platform, EFI, ACPI, PAL, SAL, FPSWA
- Resource Allocation
  - Scheduling, context switching, memory, I/Os…

Required for Customer Experience

Lesson Learned: vi ≠ System Management Tool

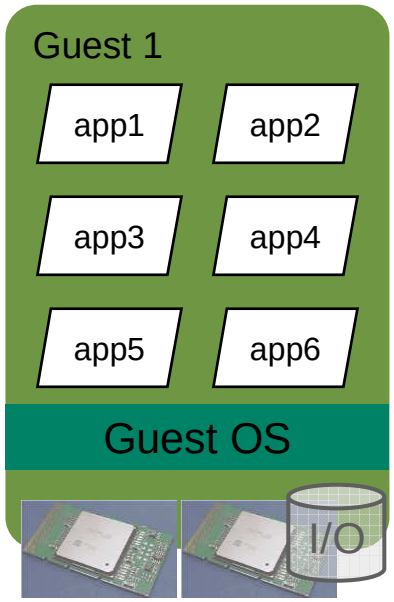
- User Interface
  - Simulate the experience of a real system
- Configuration and Management Tools (CLI)
  - Create, modify, monitor virtual machines
- Programmatic Access (API)
  - Enables Web-based management tools
- Security, Robustness, Licensing
  - OS stress test, certifications, compatibility, …
- Performance
  - Good, Predictable, Scalable, Measurable

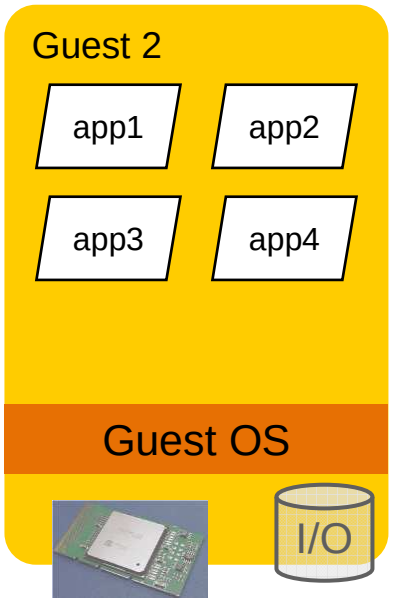
EPIC Challenge #1
Context Switching

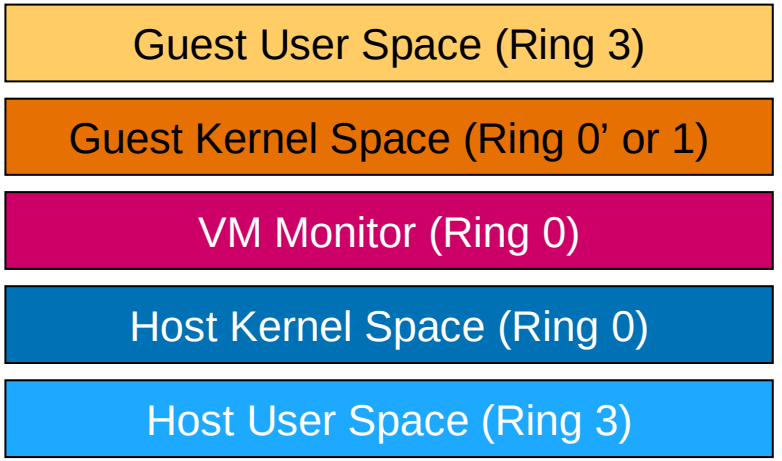


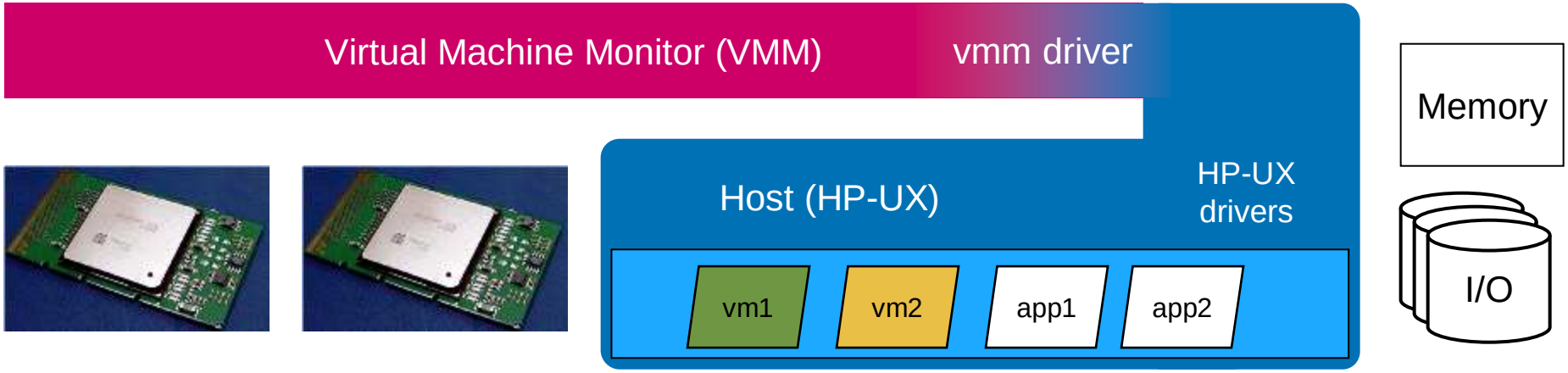
So Many Contexts

Also known as: Bit Leaks Considered Evil









So Many Transitions

Needed Some Creativity Just to Name Them…

Guest Interrupt

Guest User Space (Ring 3)

Guest Kernel Space (Ring 0' or 1)

To/from

VM Monitor Low Level (Ring 0 Asm)

Bubble-up / Bubble-down

VM Monitor Low Level (Ring 0 Asm) To/from VM Monitor High Level (Ring 0 C code)

Transmogrification, aka Xmog

VM Monitor (Ring 0) To/from VM Driver in Host Kernel (Ring 0)

Host System Call (ioctl)

VM Driver (Ring 0) To/from VM App in Host User Space (Ring 3)

Context switch

Host user-space context (Ring 3)

Host kernel-space context (Ring 0)

To/from

Host user-space context (Ring 3)

Host kernel-space context (Ring 0)

Other transitions are generally forbidden

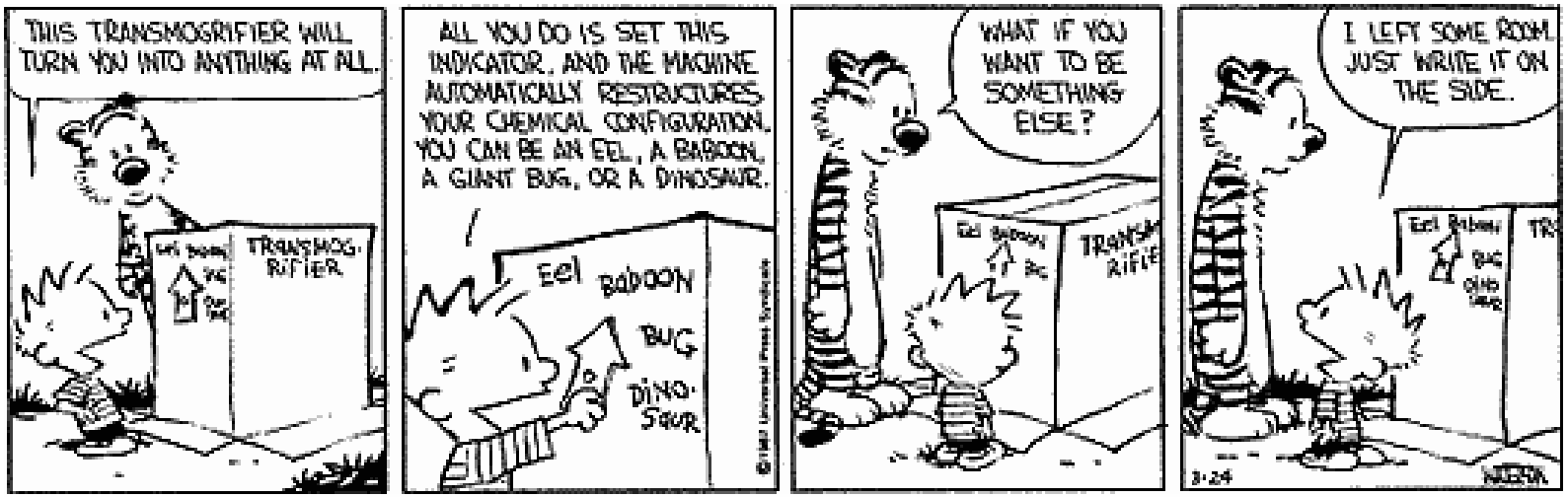
Piecemeal context switching

Why copy 8K of data if you can avoid it?

- The Itanium context is too large
  - 143 int, 128 FP, 128 CR, 128 AR, 64 predicates, +++
- Lazy context management
  - Register stack engine (RSE) spills and fills automatically
  - Floating point state large, but not always used
  - Software conventions for applications and kernel
- System state is spread all over the place
  - Stacked registers torn between guest and monitor stacks
  - Floating point state live in monitor registers
  - Scratch registers saved early, but not preserved ones

Cultural Digression

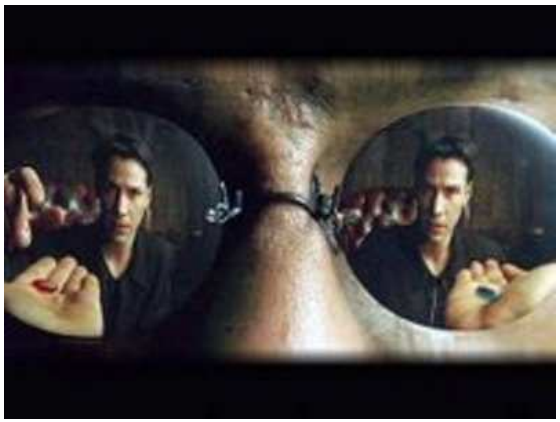
Why “Transmogrification”?



Suggested explanations:

- Allows HP-UX to “transmogrify” into Linux, Windows, a baboon, whatever…
- It’s about as reassuring as the original cardboard box technology

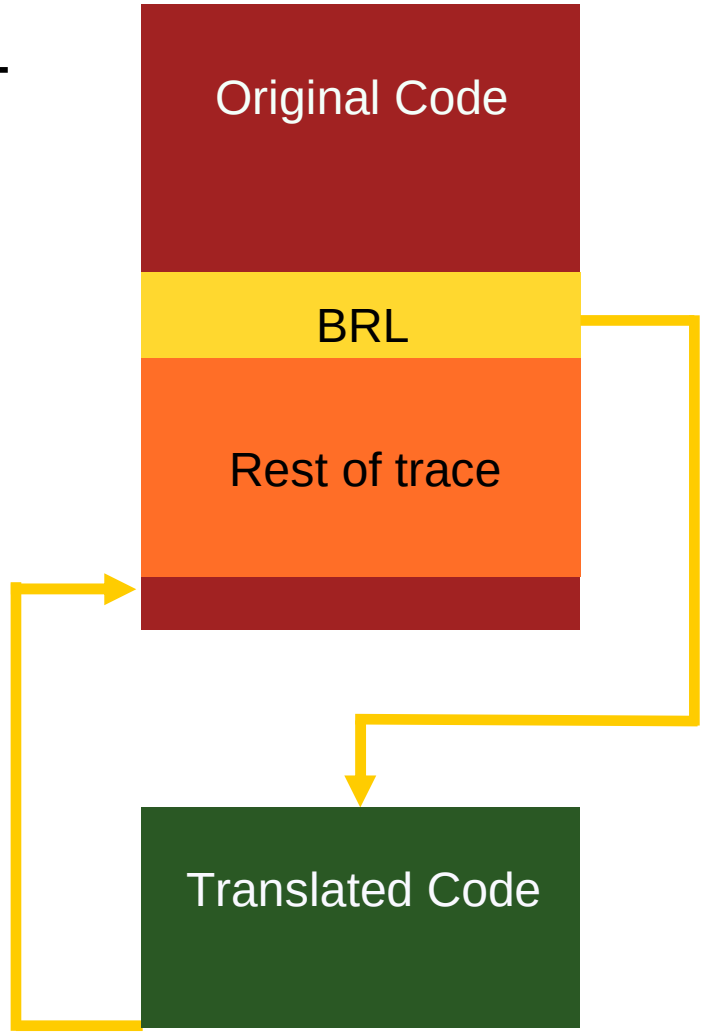
EPIC Challenge #2
Instruction Emulation



Binary Translator

“Adding one level of indirection” design school

- Patch whole bundles with BRL
- Multiple priv-ops at once
- The rest executes in place



Challenges:

MP: No atomic 16-byte store

Interrupts in translated code

RFI to middle of patched bundle

BRL is IP-relative: aliasing issue

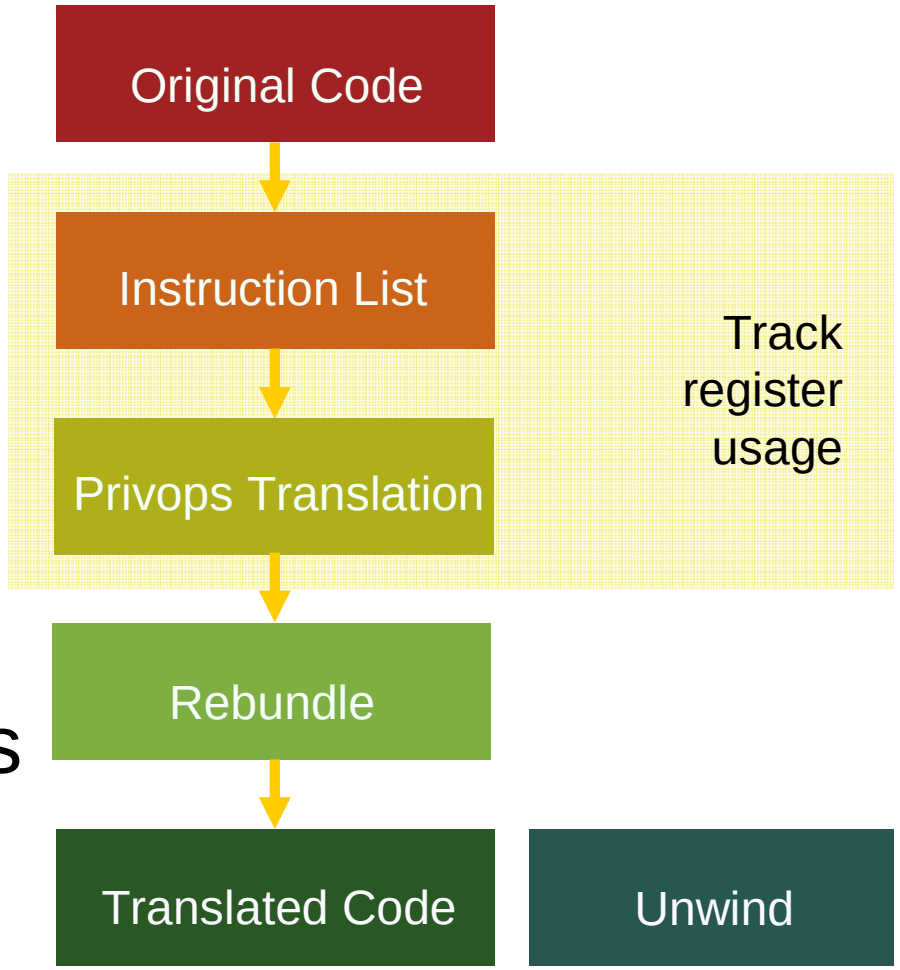
BRL on Itanium 2 only

Branch prediction

Translation = Break, Rebuild

Not unlike the Star Trek transporter

- Unbundle
- Decode
- Track Register
- Translate or Copy
- Patch Branches
- Create Unwind Tables
- Rebundle



Three Kinds of Instruction Emulation

Too complex? Add *another* level of indirection

- In-line Emulation
  - Generate sequences of replacement instructions
  - Very efficient for operations like virtual control register accesses
  - Doesn’t work when many side effects result from the instruction
- Calls to VMM gateway page
  - Out-of-line code called by the translated code
  - Restrictions (needs a stack frame, must save registers before call)
- VMM breaks
  - Both low-level (assembly) and high-level (bubble-up to C code)
  - Used for instructions with complicated side effects (TPA, ITC, RFI)
  - Slower. In the worst case, context switch to the Host!

Instruction Emulation Challenges

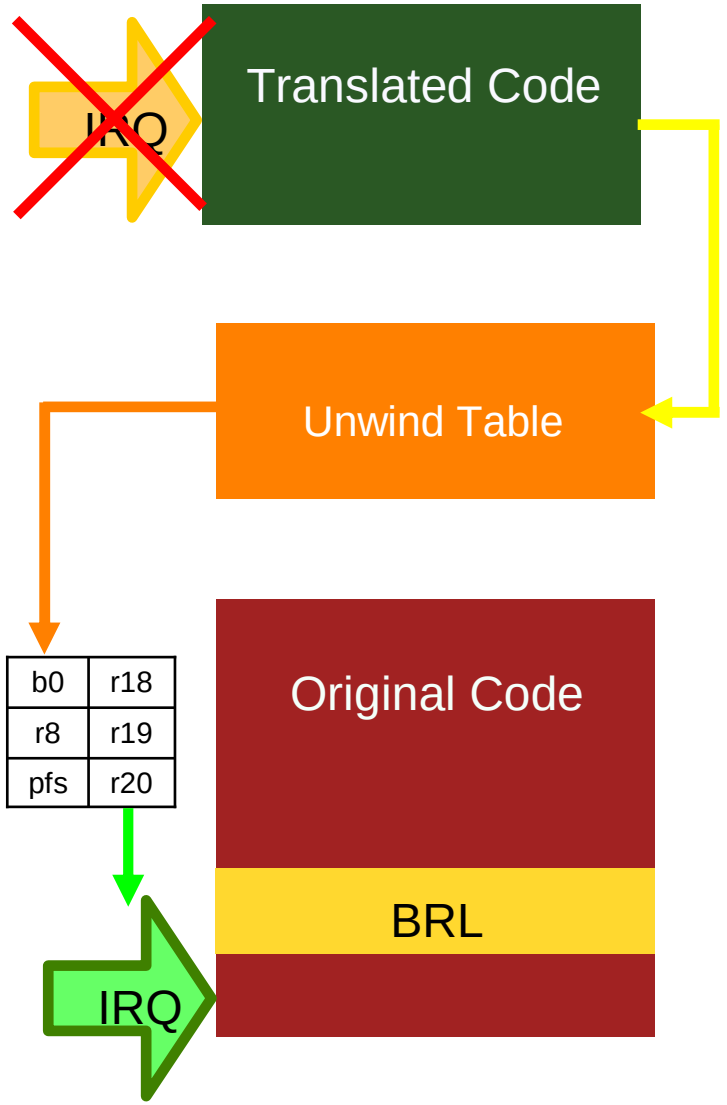
Lies, Damn Lies and Binary Translation

- Guest endianness
  - Changing endianness is not a privileged operation
  - Loads and stores to monitor state affected by guest endianness
  - Need to check at trace entry and to detect PSR.be changes
  - Performance impact, even if rarely used
- Loads and stores to I/O devices
  - Cannot be pre-decoded
  - One instruction often used for multiple addresses
- Register usage (*water, water everywhere*…)
  - You can't save a register without a register...
  - Need to be smart and borrow from context whenever possible
  - Two strategies: floating-point registers, break instructions
- Privilege sensitivity
  - Instructions that do not trap (without VT-i): thash, ttag, cover
  - Instructions using state we modified (e.g. floating-point registers)
- Self-modifying code, memory aliasing, …

Interruptions in translated code

Scotty, pull him from that pattern buffer!

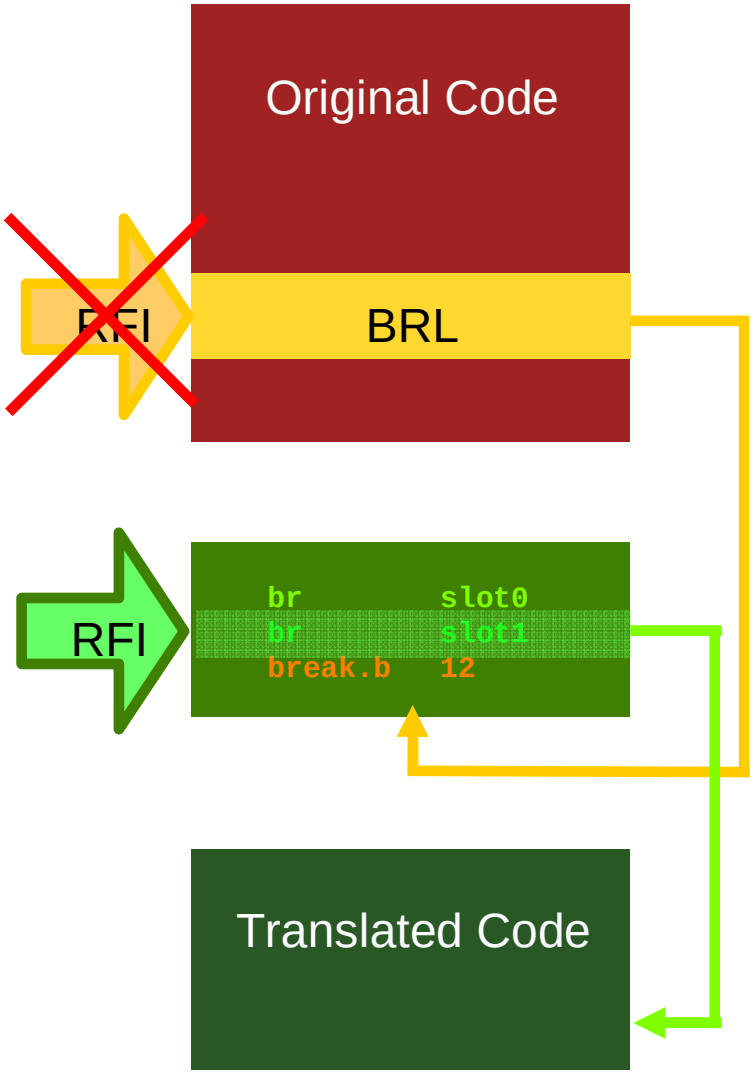
- Interruption in a trace
  - Reconstruct a valid instruction pointer
  - Restore register values if borrowed
- Unwind tables
  - Incremental, 32-bit per slot
  - Describe register usage, branches
- Interruption mid-instruction
  - Defer external interrupts until trace exit
  - Guarantee no faults or traps from our code
- Hide trace addresses
  - Don’t leave a trace address in IIP



Return from Interruption

Whoa! Déjà vu

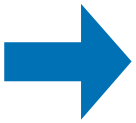
- RFI in guest code
  - May RFI to patched instruction
  - Slots in BRL don't match
  - RFI to BBB bundle instead
  - BBB bundle at BRL target-0x10
- Consequences
  - Up to 3 traces per slot
  - Lazy translation (break in BBB)



Pattern Matching

When you lose… *cheat*!

```
REPLACE:
{.mmi
            srlz.d
            mov         r36=cr.tpr
            mov         r34=b0
}
{.mlx
FAULT1      lfetch      [r32]
            movl        r25=VAR1
}
{.mmi
            adds        r30=12,r32
            adds        r39=188,r35
            mov         r40=r32;;
}
{.mmi
FAULT2      ld4         r25=[r25]
FAULT3      ld4         r26=[r39]
            mov         r37=pr;;
}
{.mmi
            cmp4.ne     p1,p0=0,r25
            cmp4.eq     p2,p0=0,r26
            adds        r29=8,r32;;
}
```



```
WITH:
            VP=r42
            br.sptk     PAT(slot0)
            br.spnt     PAT(slot1)
            br.spnt     PAT(slot2)

PAT(slot0):
//          srlz.d
PAT(slot1):
//          mov         r36=cr.tpr
PAT(slot2):
            adds        r30=12,r32
            movl.set    VP=VCPU(X_endianKey)
            ;;
            ld8         r25=[VP],.to GE_TPR
            adds        r39=188,r35
            mov         r34=b0
            ;;
            ld8         r36=[VP],.to GE_PSR
            mov         r37=pr
            cmp.ne      p4,p0=.int(ENDIAN_KEY),r25
            ;;
```

Concept Programming Applied to HP Integrity VM Code…

Virtualized…

Practical Itanium Virtualization

I once had a dream…

- From wild idea to successful product
  - A small team can make a difference
  - Thing big, know where you are going, one step at a time
  - Focus on what is needed, make it beautiful if you can
- So many interesting technical challenges
  - Binary translation, memory management, I/O drivers, …
  - Possible thanks to “The best team we ever worked for”
- Non-technical challenges also matter
  - Building a good team is essential (from small to bigger)
  - Making a business case, putting career at risk, timing
  - Working remotely is fashionable, but strenuous

What Now?

The best way to predict the future…

- Software: Workload Management
  - Objectives: Costs, performance, power, agility
  - Tools: On-line migration, capacity planning, deployment
- Hardware: Architected Virtualization
  - CPUs and chipsets now have virtualization built-in
  - The fabrics themselves: SAN, VirtualConnect, VLAN,…
- Standardization vs. Balkanization
  - The biggest threat to Itanium: “x86 everywhere”
  - But: custom chips, accelerators, clusters of PS/3…
  - A very large number of Linux variants, filling niches
  - Same thing happening in VM space with Xen or KVM


Thank
You


invent